Preface
Arrogance and ambition: In praise of outsider AI

- AGI is outsider AI
- Outsiders do better at
  - thinking for themselves
  - questioning the status quo
  - seeing the obvious
- AGI is on the hairy edge between arrogance and ambition
seeing what is obvious, and therefore invisible

- The discovery of gravity, by Isaac Newton
- The discovery of air/vacuum
- The discovery of reinforcement learning, by Harry Klopf, in the 1970s
  - RL was born seemingly in rude arrogance, by a total outsider

Harry Klopf
1941–1997
My conclusion

- Outsider AI is good
- Questioning the problem is often the best way to make the important advances
Personal perspective

• There is a science of mind that is neither natural science nor applications technology
  - as in, e.g., Marr’s “computational theory”

• “Minds” can be defined as things more usefully thought of in terms of goals than of mechanisms
  - goals can be well thought of as rewards

• Reinforcement learning is part of the beginning of a science of mind
Toward Learning Human-level Predictive Knowledge

Rich Sutton
Reinforcement Learning and Artificial Intelligence Lab
University of Alberta, Canada

with thanks to
Hamid Maei, Csaba Szepesvari, Doina Precup, Shalabh Bhatnagar,
David Silver, Michael Delp, Eric Weiwiора, and Mark Ring
The problem

- *How we can know lots of stuff* about how the world works and what we can do, and apply it efficiently to maximize reward
- We know so much! So much sensori-motor stuff
- How can we relate higher-level knowledge to the low-level sensorimotor stuff?
- How can it all be organized and maintained? What are the principles?
Much of mind is about prediction

- *Perception* and *state representation* can be thought of as making predictions.

- *Models the world* and *cause and effect* can be thought of in terms of predictions.

- *Planning* can be thought of as composing predictions to anticipate possible futures, and then choosing among them.

- *Predictions are the coin of the mental realm*
World knowledge

- Much knowledge is *about the world*
  - but not all, e.g., mathematics, memories
- All knowledge about the world is *predictive*, meaning that it can be translated into statements about future experience
- Such statements are the content of the knowledge
- They make the knowledge potentially verifiable
Predictions can be more powerful than you think

- Not just a “saying before” of what the sensory signals will be
- All scientific knowledge can be expressed as predictions
- Predictions can be about the outcomes of extended courses of behavior (options)
- All the little things you know can be well thought of as prediction
Predictions are signals

- They have a value that varies from time to time
- Their statement about the world is always relative to the current time
- Each prediction is a signal, a time series
Predictions have two parts: a “question” and an “answer”

- **Question**: will the sun rise tomorrow?  
  **Answer**: yes

- **Question**: will i win this backgammon game?  
  **Answer**: probability 0.6

- The question is future oriented whereas the answer is strictly a function of the past

- The question is the semantics of the prediction; it is needed for learning and verification, but not for performance
Questions are more important, more mysterious, more often overlooked; answers are relatively straightforward.

• E.g., flipping a coin
  - Question: what is the probability of heads
  - Answer: 0.5

• How to represent *flipping*, *coin*, and *heads*?
The robot and the battery charger

Would a person say that's a battery charger?

Would try-to-dock succeed?

semantics grounded in human judgments

semantics grounded in prediction

Only the predictive question can be autonomously verified
Desiderata: Predictive knowledge should be

1. Useful, e.g., for planning
2. Expressive (powerful, abstract, human-level)
3. Autonomously verifiable
4. Suitable for efficient learning with approximations

The theory of options from 1999 satisfies all but the last. Today, instead, we will talk of value functions
• The problem of learning predictive knowledge

• Value functions have been key to RL

• General value functions may be key to the problem of human-level predictive knowledge
  - many things work out neatly

• But learning must be off-policy and use function approximation; using the new GQ algorithm this can, at last, be done efficiently
Value functions
Value functions

- Value functions provide moment-to-moment estimates of the total future reward an AI agent can expect to receive.
- They are predictions!
  - **Question**: how much total reward will I receive?  
    **Answer**: a single real number (at each time).

- There are both true value functions and estimated value functions.
Target value functions

- A state-value function maps states to values, given a policy

\[ V^\pi(s) = E \left[ r_1 + \gamma r_2 + \gamma^2 r_3 + \cdots \mid s_0 = s, a_0:\infty \sim \pi \right] \]

- An action-value function is the same except it commits to the first action as well

\[ Q^\pi(s, a) = E \left[ r_1 + \gamma r_2 + \gamma^2 r_3 + \cdots \mid s_0 = s, a_0 = a, a_1:\infty \sim \pi \right] \]
TD-Gammon

Start with a random Network

Play millions of games against itself

Learn a value function from this simulated experience

Six weeks later it’s the best player of backgammon in the world

Tesauro, 1992-1995
The Mountain Car Problem

Minimum-Time-to-Goal Problem

SITUATIONS: car’s position and velocity

ACTIONS: three thrusts: forward, reverse, none

REWARDS: always -1 until car reaches the goal

No Discounting

Moore, 1990
Value Functions Learned while solving the Mountain Car problem

Minimize Time-to-Goal

Value = estimated time to goal
The value-function hypothesis

All efficient methods for solving sequential decision problems estimate value functions as an intermediate step
Value-function approximation

- Value-function learning is sometimes done in a table-lookup context - where every state is distinct and treated totally separately.
- But really, to be powerful, we must generalize between states.
  - The same state never occurs twice.
- We use parameterized function approximators, and learn the parameters, aka weights.
For example, linear value-function approximation in Computer Go

\[ f(s) = w^T f \]

\[ w = [0.1, -2, 0, 0.5, 0, -0.4] \]

\[ f(s) = [0, 1, 0, 0, 0, 1, 0] \]

\[ 10^{35} \text{ states} \]

\[ 10^6 \text{ binary features and weights} \]
The problem of learning predictive knowledge

Value functions have been key to RL

General value functions may be key to the problem of human-level predictive knowledge
- many things work out neatly

But learning must be off-policy and use function approximation; using the new GQ algorithm this can, at last, be done efficiently
General value functions
Multiple value functions as a knowledge rep’n language

- Value functions are predictions, long-term predictions
- They predict reward, but couldn’t we pretend anything is reward, and learn a value function for getting it?
  - if i open it the fridge, will i see a beer?
  - if i take out my wallet, will i see any euros?
  - if i do what i usually do, will the sun rise tomorrow?
Termination tricks

- It is common in RL problems to have terminal values, e.g., $+1/-1$ for winning or losing a game.

- Termination means complete instantaneous discounting, together with a special terminal value that can depend on the state.

- We can use pretend termination to escape from exponential discounting.
  
  - Will Sweden or Norway be the next to win a gold medal? (no discounting until one wins, then complete discounting/termination)
General value functions

- Conventional value functions are predictions wrt the rewards, discount, and terminal values of the problem, for a given policy

\[
Q^\pi(s, a) = \mathbb{E}[r_1 + \gamma r_2 + \gamma^2 r_3 + \cdots | s_0 = s, a_0 = a, a_1:\infty \sim \pi]
\]

\[
= \mathbb{E}[r_1 + \cdots + r_k + z_k | s_0 = s, a_0 = a, a_1:k \sim \pi, k \sim \gamma]
\]

- General value functions are predictions wrt to four given functions

\[
Q^{\pi, r, \gamma, z}(s, a) = \mathbb{E}[r(s_1) + \cdots + r(s_k) + z(s_k) | s_0 = s, a_0 = a, a_1:k \sim \pi, k \sim \gamma]
\]

these four functions define the semantics of the prediction
General value functions

\[ Q^\pi, r, \gamma, z (s, a) = \mathbb{E}[r(s_1) + \cdots + r(s_k) + z(s_k) \mid s_0 = s, a_0 = a, a_{1:k} \sim \pi, k \sim \gamma] \]

these four functions define the semantics of the prediction

\[
\begin{align*}
\text{policy} & \quad \pi : A \times S \rightarrow [0, 1] \\
\text{reward} & \quad r : S \rightarrow \mathbb{R} \\
\text{termination} & \quad \gamma : S \rightarrow [0, 1] \\
\text{terminal value} & \quad z : S \rightarrow \mathbb{R}
\end{align*}
\]

These are the prediction’s question!
General value functions

\[ Q^{\pi, r, \gamma, z}(s, a) = \mathbb{E}[r(s_1) + \cdots + r(s_k) + z(s_k) \mid s_0 = s, a_0 = a, a_{1:k} \sim \pi, k \sim \gamma] \]

There is also a Bellman equation for GVFs:

\[ Q^{\pi, r, \gamma, z}(s, a) = \sum_{s'} P(s' \mid s, a) \left[ r(s') + \gamma(s') \sum_{a'} \pi(a' \mid s') Q^{\pi, r, \gamma, z}(s', a') + (1 - \gamma(s')) z(s') \right] \]
General value functions

\[ Q^{\pi,r,\gamma,z}(s,a) = \mathbb{E}[r(s_1) + \cdots + r(s_k) + z(s_k) \mid s_0 = s, a_0 = a, a_{1:k} \sim \pi, k \sim \gamma] \]

There is also a Bellman equation for GVFs:

\[ Q^{\pi,r,\gamma,z}(s,a) = \sum_{s'} P(s'\mid s,a) \left[ r(s') + \gamma(s') \sum_{a'} \pi(a'\mid s')Q^{\pi,r,\gamma,z}(s',a') + (1 - \gamma(s'))z(s') \right] \]

and a TD (temporal difference) error:

\[ \delta_t = r(s_{t+1}) + \gamma(s_{t+1}) \sum_{a'} \pi(a'\mid s_{t+1})\hat{Q}(s_{t+1},a') + (1 - \gamma(s_{t+1}))z(s_{t+1}) - \hat{Q}(s_t, a_t) \]

thus there will be simple, cheap, TD learning algorithms
General value functions

and a TD (temporal difference) error:

$$\delta_t = r(s_{t+1}) + \gamma(s_{t+1}) \sum_{a'} \pi(a'|s_{t+1}) \hat{Q}(s_{t+1}, a') + (1 - \gamma(s_{t+1})) \pi(z_{t+1}) - \hat{Q}(s_t, a_t)$$

thus there will be simple, cheap, TD learning algorithms

e.g., tabular GQ:

$$\Delta \hat{Q}(s_t, a_t) = \alpha \delta_t \quad \text{(generalizes tabular Q-learning)}$$

with function approximation, there are also simple methods, but just like for regular value functions, conventional methods are stable only for the linear, on-policy case

if we can’t learn off-policy it spoils everything!
The problem of learning predictive knowledge

Value functions have been key to RL

General value functions may be key to the problem of human-level predictive knowledge
- many things work out neatly

But learning must be off-policy and use function approximation; using the new GQ algorithm this can, at last, be done efficiently
Off-policy learning with GQ
Off-policy learning

- Off-policy learning is learning about a policy different than that used to generate actions.
  - Most often arises when learning an optimal policy while following an exploratory policy, or from a pre-collected data set.
- We are envisioning learning about many policies, so it will have to be off-policy (or else the policies have to take turns 😞).
- Off-policy learning is roughly the same as learning from incomplete trajectories.
Parallel prediction demons

- We should be able to learn lots of value functions at once, in parallel
  - we call them \textit{parallel prediction demons}

- \textit{Every} demon should be able to learn on every step

- This has always been the promise of off-policy temporal-difference learning
But this promise has been unfulfilled

- There has been no practical algorithm for parallel prediction learning
  - Previous methods were too complex (LSTD, iLSTD), restricted to table lookup (Q-learning), not parallel (Monte Carlo, Sarsa), too slow (importance sampling), or had weak approximators (averaging)
Until now

• Now, for the first time, it is practical and straightforward to do massive, in parallel, prediction learning

• With new gradient-based TD algorithms
  - GTD, TDC (NIPS-08, ICML-09, NIPS-09)
  - GQ(\(\lambda\)) (AGI-10)
Baird’s counterexample

- A simple Markov chain
- Linear FA, all rewards zero
- Deterministic, expectation-based full backups (as in DP)
- Each state updated once per sweep (as in DP)
- Weights can diverge to ±∞
Linear GQ(0)

Linear approximation of the GVF:

\[ \hat{Q}(s, a) = w^\top f(s, a) \approx Q^{\pi, r, \gamma, z}(s, a) \]

Learning:

\[ \Delta w_t = \alpha \delta_t f(s_t, a_t) - \alpha \gamma(s_{t+1})(v_t^\top f(s_t, a_t)) \sum_a \pi(a|s_{t+1}) f(s_{t+1}, a) \]

\[ \Delta v_t = \beta (\delta_t - v_t^\top f(s_t, a_t)) f(s_t, a_t) \]

It’s one more step bring in eligibility traces...
Linear GQ(λ)

Linear approximation of the GVF:

\[ \hat{Q}(s, a) = w^\top f(s, a) \approx Q^{\pi, r, \gamma, \lambda}(s, a) \]

Learning:

\[ \Delta w_t = \alpha \delta_t e_t - \alpha \gamma(s_{t+1})(v_t^\top e_t) \sum_a \pi(a|s_{t+1})f(s_{t+1}, a) \]

\[ \Delta v_t = \beta (\delta_t e_t - v_t^\top f(s_t, a_t)f(s_t, a_t)) \]

\[ e_t = \gamma(s_t)\lambda(s_t)\frac{\pi(a_t|s_t)}{b(a_t|s_t)}e_{t-1} + f(s_t, a_t) \]

The behavior policy, the policy actually picking the actions.

A separate set of weights used only for learning.
Stability and convergence theorem for GQ(\(\lambda\))

There exists a projected-Bellman-error objective function

\[
J(w) = \left\| \hat{Q}_w - \Pi T^{\pi, r, \gamma, z} \hat{Q}_w \right\|_b^2
\]

such that

\[
E_b[\Delta w] = -\alpha \nabla_w J(w)
\]

which guarantees convergence to \(J(w) = 0\)
(under step-size conditions)
Further results with new gradient-descent TD methods

- Convergence with nonlinear function approximators (e.g., neural networks)
- Empirical on-policy learning rate comparable to that of linear and nonlinear TD on 9x9 Computer Go
- First convergence result for the control case (changing target policy $\pi$)
- One-time-scale proofs of convergence for all algorithms; constant second step-size
Application to computational curiosity

• Imagine one million prediction demons, all learning in parallel
  - for various random or cleverly chosen GVF$s$
• Imagine each can measure its learning progress
• Use the sum-total learning progress as intrinsic reward to direct the behavior policy
• Weed and refine the set of demons, then repeat
The problem of learning predictive knowledge
Value functions have been key to RL
General value functions may be key to the problem of human-level predictive knowledge
- many things work out neatly
But learning must be off-policy and use function approximation; using the new GQ algorithm this can, at last, be done efficiently
Conclusions

• The new gradient TD algorithms are a breakthrough in RL (two open probs solved)

• Function approximation in RL is now nearly as straightforward as supervised learning

• General value functions turn out to be a very expressive knowledge rep’n language
  - their underlying Bellman equation makes (TD) learning them computationally efficient
  - they take us from values to knowledge very economically, with few new ideas
What is new?

- Efficient off-policy learning is new
- General value functions are a new, simpler way to present the ideas of options and option models
- Applications to computational curiosity are new and ongoing
- Applications to representation change and state discovery are starting
In its ambitions, AGI should be...

(a) Arrogant and abrasive
(b) Audacious
(c) Appropriate
(d) All of the above

✓ (d) All of the above
• thank you for your attention